Homework 4

CMSY-199, Fall 2013

Upload your solution to the Canvas course website as a zip archive file prior to the start of class on Monday, November 4.

1. Create a Java class called `Matrix` which will be used to represent a rectangular array of numbers and perform basic matrix operations. If you are not familiar with matrices, please read the sections beginning on pages 1-6 and 2-2 of the MATLAB Primer which can be found here:
   http://www.mathworks.com/help/pdf_doc/matlab/getstart.pdf

2. The `Matrix` class should have a single instance variable which is a two-dimensional array of type `double` and a constructor which takes a two-dimensional array as an argument to set its instance variable.

3. Add a single `set` method which takes two `int` values and one `double` as arguments which specify the row, column, and value of the element to set. Add a single `get` method which takes two `int` values which specify the row and column of the element to get. Note that for the `Matrix` class, rows and columns should 1-based even though the array backing it is 0-based.

4. Add the following `static` methods to the `Matrix` class.

Name	**Parameters**	**Return Type**	**Description**
`ones`	`int row, int column`	`Matrix`	Return a new matrix with the specified number of rows and columns - each element has value 1
`zeros`	`int row, int column`	`Matrix`	Return a new matrix with the specified number of rows and columns - each element has value 0
`rand`	`int row, int column`	`Matrix`	Return a new matrix with the specified number of rows and columns - each element is a uniformly distributed random value
`randn`	`int row, int column`	`Matrix`	Return a new matrix with the specified number of rows and columns - each element is a normally distributed random value
`concat`	`Matrix a, Matrix b`	`Matrix`	Return a new matrix which is the *horizontal* concatenation of matrices a and b

5. Add the following methods to the `Matrix` class.

Name	Parameters	Return Type	Description
`add`	`Matrix b`	`void`	Perform element-wise addition on `this` matrix using matrix b
`subtract`	`Matrix b`	`void`	Perform element-wise subtraction on `this` matrix using matrix b
`pow`	`double n`	`void`	Raise each element of `this` matrix to the n-power
`transpose`		`void`	Transpose `this` matrix (swap rows and columns)
`sin`		`void`	Take the sine of each element in `this` matrix
`max`		`double`	Return the element with the maximum value in `this` matrix
`sum`		`double[]`	Return an array which contains the sum of each column of `this` matrix
`diag`		`double[]`	Return an array which contains diagonal of `this` matrix
`equals`	`Object o`	`boolean`	Return true if `o` is an instance of `Matrix` and each element is equal to the elements in `this`
`toString`		`String`	Return a `String` object which contains the elements of `this` in tabular fashion by row and column

6. Create a *driver* class called `MatrixTest` which has a `main` method to test the functionality of the `Matrix` class.